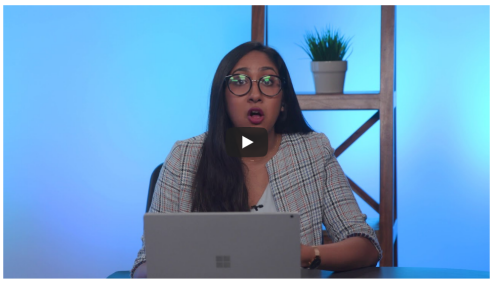
Image Data

Images are another example of a data type that is commonly used as input in machine learning problems—but that isn't initially in numerical format. So, how do we represent an image as numbers? Let's have a look.



Taking a Closer Look at Image Data

Let's look a little closer at how an image can be encoded numerically. If you zoom in on an image far enough, you can see that it consists of small tiles, called *pixels*:

The color of each pixel is represented with a set of values:

- In **grayscale images**, each pixel can be represented by a **single** number, which typically ranges from 0 to 255. This value determines how dark the pixel appears (e.g., `0` is black, while `255` is bright white).
- In **colored images**, each pixel can be represented by a vector of **three** numbers (each ranging from 0 to 255) for the three primary color channels: red, green, and blue. These three red, green, and blue (RGB) values are used together to decide the color of that pixel. For example, purple might be represented as `128, 0, 128` (a mix of moderately intense red and blue, with no green).

The number of channels required to represent the color is known as the **color depth** or simply **depth**. With an *RGB image*, `depth = 3`, because there are three channels (Red, Green, and Blue). In contrast, a grayscale image has `depth = 1`, because there is only one channel.

Encoding an Image

Let's now talk about how we can use this data to encode an image. We need to know the following three things about an image to reproduce it:

- Horizontal position of each pixel
- Vertical position of each pixel
- Color of each pixel

Thus, we can fully encode an image numerically by using a vector with three dimensions. The size of the vector required for any given image would be the `height * width * depth` of that image.

QUESTION 1 OF 2

Assume this figure is the *numerical representation* of an **RGB image** in the red channel:

25	5	110	34
71	207	48	99
18	156	60	7
55	39	170	32

Each of the squares represents one pixel and the value in the square is the pixel value.

Which of these statements is **incorrect**?

- This image can be encoded by a vector with the dimension of `4*4*3`
- ✓ The total number of pixels in this image is 48
- The numerical representation of the image in the **green channel** has the dimension of `4*4`
- The image has uniform aspect ratio but may need to be normalized.

SUBMIT

QUESTION 2 OF 2

There is a square shaped **RGB image** that consists of 900 pixels. Which of the following statements are correct?

- Without any preprocessing, the image can be encoded by a 3-dimension vector with the dimension `45*20*3`
- ✓ This image has a dimension of `30*30`
- If the image is cropped to half of the original size, it can be encoded by a vector with the dimension `15*15*2`
- ✓ If the image is converted to grayscale, it can be encoded by a vector with the dimension `30*30*1`

SUBMIT

Other Preprocessing Steps

In addition to encoding an image numerically, we may also need to do some other preprocessing steps. Generally, we would want to ensure that the input images have a *uniform aspect ratio* (e.g., by making sure all of the input images are square in shape) and are *normalized* (e.g. subtract mean pixel value in a channel from each pixel value in that channel). Some other preprocessing operations we might want to do to clean the input images include rotation, cropping, resizing, denoising, and centering the image.

NEXT